



03CO
0400
#4

500.39920X00

THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s): A. NAKAMURA, ET AL.
Serial No.: 09 / 811,404
Filed: MARCH 20, 2001
Title: "PROGRAM CONTROL SYSTEM AND PROGRAM CONTROL METHOD".

LETTER CLAIMING RIGHT OF PRIORITY

Honorable Commissioner of
Patents and Trademarks
Washington, D.C. 20231

APRIL 23, 2001

Sir:

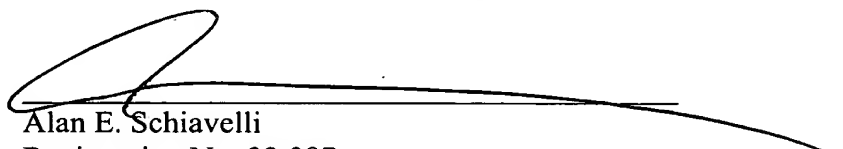
Under the provisions of 35 USC 119 and 37 CFR 1.55, the applicant(s) hereby claim(s)
the right of priority based on:

Japanese Patent Application No. 2000 - 164719
Filed: MAY 30, 2000

A certified copy of said Japanese Patent Application is attached.

Respectfully submitted,

ANTONELLI, TERRY, STOUT & KRAUS, LLP


Alan E. Schiavelli
Registration No. 32,087

AES/rp
Attachment



日本国特許庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日

Date of Application:

2000年 5月30日

出願番号

Application Number:

特願2000-164719

出願人

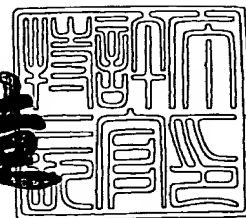
Applicant(s):

株式会社日立製作所

2001年 4月 6日

特許庁長官
Commissioner,
Patent Office

及川耕造



出証番号 出証特2001-3025847

【書類名】 特許願

【整理番号】 K00009381

【提出日】 平成12年 5月30日

【あて先】 特許庁長官殿

【国際特許分類】 G06F 19/00

【請求項の数】 4

【発明者】

 【住所又は居所】 神奈川県横浜市戸塚区戸塚町 5 0 3 0 番地 株式会社日立製作所 ソフトウェア事業部内

 【氏名】 中村 敦

【発明者】

 【住所又は居所】 神奈川県横浜市戸塚区戸塚町 5 0 3 0 番地 株式会社日立製作所 ソフトウェア事業部内

 【氏名】 由井 仁

【発明者】

 【住所又は居所】 神奈川県川崎市幸区鹿島田 8 9 0 番地 株式会社日立製作所 ビジネスソリューション開発本部内

 【氏名】 吉田 順

【特許出願人】

 【識別番号】 000005108

 【氏名又は名称】 株式会社日立製作所

【代理人】

 【識別番号】 100075096

 【弁理士】

 【氏名又は名称】 作田 康夫

【手数料の表示】

 【予納台帳番号】 013088

 【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 プログラム制御システム及びプログラム制御方法

【特許請求の範囲】

【請求項 1】

第 1 のアプリケーションの実行結果を用いて第 2 のアプリケーションの実行を制御するプログラム制御システムにおいて、
前記第 1 のアプリケーションの実行結果を前記第 2 のアプリケーションが使用するために、前記実行結果を変換する変換部と、
変換された前記実行結果を用いて前記第 2 のアプリケーションを実行するプログラム実行部とを有するプログラム制御システム。

【請求項 2】

第 1 のアプリケーションの実行結果を用いて第 2 のアプリケーションの実行を制御するプログラム制御システムにおいて、

前記第 1 のアプリケーションの実行結果を前記第 2 のアプリケーションが使用するために、前記実行結果を変換する変換プログラムと、

前記第 1 のアプリケーションの実行終了に基づいて前記変換プログラムを動作させる手段と、

前記第 1 のアプリケーションの実行終了に基づいて前記第 2 のアプリケーションを動作させる手段とを有するプログラム制御システム。

【請求項 3】

請求項 1 のプログラム制御システムにおいて、

前記変換プログラムを動作させる手段と、前記第 2 のアプリケーションを動作させる手段は、前記第 1 のアプリケーションの実行終了を関しするワークフロー管理プログラムによって制御されるプログラム制御システム。

【請求項 4】

第 1 のアプリケーションの実行結果を用いて第 2 のアプリケーションの実行を制御するプログラム制御方法であって、

前記第 1 のアプリケーションの実行結果を前記第 2 のアプリケーションが使用するために、前記実行結果を変換する変換プログラムを有し、

前記第 1 のアプリケーションの実行終了に基づいて前記変換プログラムを動作させる処理と、

前記第 1 のアプリケーションの実行終了に基づいて前記第 2 のアプリケーションを動作させる処理とを有するプログラム制御方法。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

本発明は、プログラムの実行を制御するシステム及びプログラムの実行を制御する方法に関する。

【 0 0 0 2 】

【従来の技術】

特開 2 0 0 0 - 2 9 9 5 6 号公報には、既存のアプリケーションをワークフローシステムで制御することが開示されている。ここでは、アプリケーションのトランザクションの開始、コミット等を記述することができるセッションプログラムを含むセッションハンドルが、クライアントからの要求により、アプリケーションの処理を実行することで、ワークフロー実行中に、既存アプリケーションをワークフローシステムによって制御し、複数のクライアントでの操作を経て一つのプロセスを終了することが開示されている。

【 0 0 0 3 】

【発明が解決しようとする課題】

アプリケーションは、その結果を他のアプリケーションで使われることを前提に作られているとは限らず、一つの処理の中で、このように関連性のないアプリケーションを相互に利用することができない。

【 0 0 0 4 】

上記従来技術では、既存のアプリケーションをワークフローシステムによって制御することが開示されているが、具体的にアプリケーション間でのデータのやり取りについてまでは開示されていない。

【 0 0 0 5 】

本発明は、このような問題点に鑑みてなされたもので、関連性のないアプリケ

ーション間でデータの受け渡しを可能とするアプリケーションプログラムの実行を制御するシステム及び方法を提供することを目的とする。

【 0 0 0 6 】

更に、ワークフローで定義されているプロセス毎に異なったアプリケーションを利用することを可能とするワークフローシステム及び方法を提供することを目的とする。

【 0 0 0 7 】

これにより、簡単に、より複雑、高度な処理を実行することが可能となる。

【 0 0 0 8 】

【課題を解決するための手段】

上記目的は、第 1 のアプリケーションの実行結果を第 2 のアプリケーションで利用できるように変換する変換部と、変換された実行結果を利用して第 2 のアプリケーションを実行する実行部とを備えることにより達成される。

【 0 0 0 9 】

尚、変換部は第 1 のアプリケーションの実行結果を第 2 のアプリケーションで利用できるようにする変換プログラムによって構成されていてもよい。

【 0 0 1 0 】

【発明の実施の形態】

以下、図面を用いて説明する。

【 0 0 1 1 】

図 1 0 は、本発明のプログラム制御システムの基本構成を示したものである。プログラム制御システムは、プログラム制御装置 1 0 0 0 と、複数のアプリケーションプログラム（以下、「アプリケーション」という。） 1 0 0 3、1 0 0 4、1 0 0 5 から構成されている。プログラム制御装置 1 0 0 0 は、データ変換部 1 0 0 1、アプリケーション起動部 1 0 0 2 から構成されている。データ変換部 1 0 0 1 は、アプリケーション間でデータが利用できるように、それぞれのアプリケーションで利用されるデータの形式、内容に対応付けしてあらかじめ定義しておき、この定義に基づいてデータの形式、内容を変換するものである。アプリケーション起動部 1 0 0 2 は、各アプリケーションの初期化处理、終了時処理、ア

アプリケーションプログラムの実行順序が定義されている。

【0012】

次に、このシステムにおいて、アプリケーション1003、アプリケーション1004、アプリケーション1005の順に動作する場合について説明する。アプリケーション起動部1002によって、アプリケーション1003の初期化処理を行い、次にアプリケーション1003を実行する。アプリケーション1003の処理が終了するとアプリケーション起動部は終了時処理を実行し、アプリケーション1003のデータをデータ変換部1001へ送る。データ変換部1001は、あらかじめ対応付けされたアプリケーション1003とアプリケーション1004のデータの形式、内容に基づいてアプリケーション1003のデータをアプリケーション1004のデータへ変換する。このように変換されたデータは、再びアプリケーション起動部1002へ送らる。アプリケーション起動部1002は、アプリケーション1004の初期化処理を行い、アプリケーション1004を起動して、変換されたデータをアプリケーション1004へ入力する。アプリケーション1004の処理が終了するとアプリケーション起動部1002は終了時処理を実行し、アプリケーション1004のデータをデータ変換部1001へ送る。同様にして、データ変換を行った後、アプリケーション起動部1002でアプリケーション1005が実行される。尚、ここではアプリケーション起動部1002で、アプリケーションの実行順序が定義されている場合について説明したが、他のシステムやプログラム（アプリケーションプログラムを含む）によってアプリケーションの実行順序を制御してもよい。

【0013】

次に、ワークフローシステムに、このプログラム制御システムを適用した場合について説明する。

【0014】

図1は、本発明をワークフローシステムへ適用したシステム全体の構成を示した図である。本システムは、ワークフロー管理サーバ1、データベース9、アプリケーション13、ワークフロー管理サーバ1と複数のアプリケーション13との間でデータのやり取り等を管理する複数のアプリケーション統合装置14から

構成されている。

【 0 0 1 5 】

ワークフロー管理サーバ 1 は、作業内容及び作業の実行順序が定義されたワークフロー関連データに従って処理を実行する。また、ワークフロー管理サーバ 1 では、例えば文書データ等のフローデータが予め作業の実行順序を定義したワークフロー関連データに基づいて、ある作業から他の作業へ渡されて処理が進められていく。データベース 9 には、ワークフロー関連データ、フローデータがプロセスデータとして格納されており、ワークフロー管理サーバ 1、アプリケーション統合装置 1 4 によって読み出し及び書き込みが行われる。

【 0 0 1 6 】

アプリケーション統合装置 1 4 は、ワークフロー管理サーバ 1 との接続を管理しワークフロー管理サーバ 1 が扱うプロセスデータの管理・変換を行うサーバ接続部 2、ユーザが実装するアプリケーション 1 3 固有の処理のインターフェースを提供し、アプリケーションの実行状態をワークフロー管理サーバ 1 に通知することでワークフロー管理サーバ 1 が作業状態を変化させることができるアプリケーション管理部 3、プロセスデータマッピングテーブル 1 0、トランスレータ 1 1、トランスレータマッピング情報テーブル 1 2 から構成されている。

【 0 0 1 7 】

更に、サーバ接続部 2 は、サーバコネクション管理部 4、プロセスデータ管理部 5、データトランスレータ管理部 6 を有し、アプリケーション管理部 3 は、アプリケーション状態通知部 7、アプリケーション初期化・終了時処理実行部 8 を有している。

【 0 0 1 8 】

サーバコネクション管理部 4 は、ワークフロー管理サーバ 1 とアプリケーション 1 3 との連携にあたり、接続を開始し、接続された状態を保持する。このとき、アプリケーション 1 3 からの複数の要求に対応するためにあらかじめ複数の接続を保持しておくことを可能とする。

【 0 0 1 9 】

プロセスデータマッピング情報テーブル 1 0 は、予めワークフロー管理サーバ

1で実行される作業と、フローデータの項目とが関連付けされて格納されている。

【0020】

プロセスデータ管理部5は、プロセスマッピング情報テーブル10を利用して、データベース9に格納されたプロセスデータからフローデータを取り出し、データトランスレータ管理部6に取り出したフローデータを渡す。また、プロセスデータ管理部5は、データトランスレータ管理部6から受け取ったフローデータをワークフロー関連データに加えプロセスデータとしてデータベース9に格納する。

【0021】

データトランスレータ管理部6は、プロセスデータ管理部5から受け取ったフローデータをアプリケーション13固有のデータに変換するよう外部のデータトランスレータ11に要求する。また、データトランスレータ管理部6は、アプリケーション管理部3から受け取ったアプリケーション固有のデータをフローデータに変換するよう外部のデータトランスレータ11に要求する。トランスレータマッピング情報テーブル保持部12は、予めフローデータとアプリケーション固有のデータとの対応関係が保持されているものであり、トランスレータ11は、データトランスレータ管理部6の要求を受けるとトランスレータマッピング情報テーブル12を利用して変換を行う。すなわち、データトランスレータ管理部6、トランスレータ11、トランスレータマッピング情報テーブル12とを利用して、ワークフロー管理サーバ11で利用可能なデータをアプリケーション13で利用可能なデータに、またアプリケーション13で利用可能なデータをワークフロー管理サーバ11で利用可能なデータに変換している。

【0022】

また、1つの作業で実行されるアプリケーションが複数ある場合には、それぞれのアプリケーションで利用可能なデータ変換を行うようにすれば良い。つまり、アプリケーションAを実行した後にアプリケーションBを実行する場合、トランスレータマッピング情報テーブル12には、アプリケーションAとアプリケーションBとのデータの対応関係を格納しておけば良い。この場合には、フローデ

ータへ変換すること無しに、アプリケーションAのデータをアプリケーションBへ渡すことができるので、処理を少なくすることができ全体の処理速度が早くなる。

【0023】

尚、変換が必要なデータの例としては、一方が文字列データで表現された製品名で他方がバイナリデータで表現された製品コードのようにデータの形式が異なる場合、一方が製品の個数で他方が合計金額のようにデータの内容が異なる場合などがある。データの形式が異なる場合には、それぞれのデータの対応関係とデータの形式をトランスレータマッピング情報テーブル12に格納しておけば良い。また、データの内容が異なる場合には、データの対応関係と演算式（例えば、合計金額＝個数×単価）をトランスレータマッピング情報テーブル12に格納しておけば良い。トランスレータ11は、この格納された内容に従って変換を行う。

【0024】

アプリケーション初期化・終了時処理実行部8は、アプリケーション13固有の処理を実行するためのインターフェースを提供する。アプリケーション初期化・終了時処理実行部8は、予めアプリケーション13固有の処理を設定し、設定された順序に従って実行する。アプリケーション状態通知部7は、アプリケーション13がアプリケーション初期化・終了時処理実行部8を通して通知した開始・終了にしたがって、対応付けられた作業の状態を変更するようワークフロー管理サーバ1に要求する。このとき、要求はサーバコネクション管理部4を通じて行う。

【0025】

図3は、プロセスデータマッピング情報テーブル10の詳細を示したものである。プロセスデータマッピング情報テーブル301は、作業ごとに定義された複数のマッピングテーブル302から構成されている。それぞれのマッピングテーブルは、作業に対するフローデータの有無を示すデータ有無情報、作業で扱うデータ項目名を示したデータ項目名リスト情報、それぞれの項目のデータの形式を示したタイプ識別子リスト情報から構成されている。更に、これらの情報は、デ

データベース 9 から読み出す時に参照する「読み込み時」と、データベース 9 へ書き込む時に参照する「書き込み時」の情報から構成されている。

【 0 0 2 6 】

図 4 は、トランスレータマッピング情報テーブル 1 2 の詳細を示したものである。トランスレータマッピング情報テーブル 4 0 1 は、作業ごとに定義された複数のマッピングテーブル 4 0 2 から構成されている。それぞれのトランスレータマッピング情報テーブルは、変換の可否を示すデータ変換情報、変換前のデータの形式を示す変換元タイプリスト情報、変換後のデータの形式を示す変換先タイプリスト情報から構成されている。更に、これらの情報は、アプリケーションで処理を実行する前に参照する「入力時」と、アプリケーションで処理を実行した後に参照する「出力時」の情報から構成されている。尚、図 4 に示したトランスレータマッピング情報テーブルはデータの形式が異なる場合を示したものである。

【 0 0 2 7 】

データトランスレータ管理部 6 は、開始が要求されている作業に該当するテーブルを参照しデータトランスレータ 1 1 にフローデータを受け渡すかどうかを判断する。プロセスデータマッピングテーブル 3 0 2 の読み込み時タイプ識別子リストと、トランスレータマッピングテーブル 4 0 2 の入力時変換元タイプリストは一致している必要がある。また、トランスレータマッピングテーブル 4 0 2 の出力時変換先タイプリストと、プロセスデータマッピングテーブル 3 0 2 の書き込み時タイプ識別子リストは一致している必要がある。

【 0 0 2 8 】

図 2 は、図 1 のサーバ接続部 2 がアプリケーションを起動する場合の処理フローを示したものである。ワークフロー管理サーバ 1 がデータベース 9 に格納されたプロセスデータに従って作業を開始すると、作業の開始要求が、ワークフロー管理サーバで定義されるプロセスの各作業にあらかじめ対応付けられているサーバコネクション管理部 4 を介してプロセスデータ管理部 5 に送られる。プロセスデータ管理部 5 が作業開始要求を受け取ると（2 0 1）、プロセスデータ管理部 5 は、プロセスデータマッピング情報テーブル 1 0 から開始要求された作業に該

当するマッピング情報を取得する（202）。プロセスデータ管理部5は、取得したマッピング情報を参照して作業に該当するフローデータが存在するかを判断し、存在しなければ、アプリケーション管理部3へアプリケーション13の処理開始を要求する（209）。該当するフローデータが存在すると判断されると、作業に該当するフローデータをデータベース9から取得する（204）。その後、データトランスレータ管理部6は、作業に該当するマッピング情報をトランスレータマッピング情報テーブル12から取得する（205）。データトランスレータ管理部6は、取得したマッピング情報を参照してフローデータをアプリケーション13固有のデータに変換する必要があるかを判断する（206）。変換の必要があると判断された場合には、フローデータをデータトランスレータ11に受け渡し、データの変換を要求する（207）。変換の必要がないと判断された場合には、データトランスレータ11には変換要求を行わない。変換されたアプリケーション13固有のデータまたは変換の必要がないと判断されたフローデータをアプリケーション管理部3に渡す（208）。その後、アプリケーション管理部3へアプリケーション13の処理開始を要求する（209）。

【0029】

図11は、サーバ接続部3がアプリケーションからデータを受けたあとの処理フローを示したものである。

【0030】

アプリケーション管理部3を介してデータを受け取ると（1101）、データトランスレータ管理部6は、トランスレータマッピング情報テーブル12からマッピング情報を取得し（1102）、フローデータへ変換するか判断する（1103）。フローデータへ変換する必要がある場合には、次に処理1105を実行する。フローデータへ変換する必要がある場合には、データトランスレータにフローデータの変換を要求する（1104）。次に、データトランスレータ管理部6は、プロセスデータ管理部5へ、変換したフローデータを渡す。フローデータを渡されたプロセスデータ管理部5は、プロセスデータマッピング情報テーブル10を参照し、フローデータとワークフロー関連データとを関連付けてデータベース9へ格納する。

【0031】

図5は、図1のアプリケーション管理部3の処理フローを示したものである。アプリケーション状態通知部7が、サーバ接続部2からアプリケーション13の処理開始要求を受け取ると(501)、アプリケーション状態通知部7はアプリケーションに入力するデータが存在するかを判断する(502)。このとき、図2に示した処理208でデータが受け渡されていれば、入力データをアプリケーション初期化・終了時処理実行部8に受け渡し(503)、存在しなければ処理503は実行されない。その後、アプリケーション初期化・終了時処理実行部8に、予め設定されたアプリケーション13固有の初期化処理の実行を要求する(504)。次に、アプリケーション状態通知部7は、該当作業の実行開始済みの状態変更の要求を出す。この要求は、サーバコネクション管理部4を介してワークフロー管理サーバ1へ送られ、ワークフロー管理サーバ1は該当作業の状態を実行開始済みに変更する。アプリケーション13の処理が終了すると、アプリケーション初期化・終了時処理実行部8に予め設定されたアプリケーション13固有の終了時処理が実行され、アプリケーション状態通知部7に実行完了の要求を出し(506)、アプリケーション状態通知部7は、実行終了の状態変更要求を出す(508)。実行終了の状態変更要求は、サーバコネクション管理部4を通してワークフロー管理サーバ1へ送られる。

【0032】

尚、この図5に示した処理はプログラムとしてコンピュータにより読み取り可能なフロッピーディスク、光ディスク、磁気ディスクなどの記憶媒体に格納されているものであってもよい。図2、図11に示した処理も同様にプログラムによって与えられてもよく、このプログラムが記憶媒体に格納されているものであってもよい。

【0033】

以上説明したように本発明によれば、データの変換を可能とするので、アプリケーション間のデータの形式又は内容が異なるシステムを構築することが可能となる。これにより、既存のアプリケーションを自由に利用することが可能となるので、アプリケーションの資源を有効に利用することが可能となる。また、アプ

リケーションに依存しない部分を共通に利用することが可能となる。

【 0 0 3 4 】

また、ワークフロー管理サーバ 1 で利用されるデータとアプリケーションで利用されるデータが異なるシステム、あるいはアプリケーション間で利用されるデータが異なるシステムを構築することが可能となる。これにより、既存のアプリケーションを自由に利用することが可能となるので、アプリケーションの資源を有効に利用することが可能となる。更に、作業毎に異なった 1 つ或いは複数のアプリケーションを実行することが可能となるので、複雑な作業を実行することが可能となる。

【 0 0 3 5 】

また、アプリケーションに依存するプロセスデータマッピング情報テーブル、トランスレータマッピング情報テーブル、アプリケーション初期化処理・終了時処理以外の部分は、アプリケーションに依存しないので共通のハードウェアあるいはソフトウェアでシステムを構築あるいは提供することが可能となる。言い換えれば、アプリケーションに応じて、アプリケーション統合装置のプロセスデータマッピング情報テーブル、トランスレータマッピング情報テーブル、アプリケーション初期化処理・終了時処理を変えれば良く、汎用性に優れたシステムを構築あるいは提供することが可能となる。

【 0 0 3 6 】

以上、本発明の基本構成について説明した。以下、本発明を適用したシステムの利用例をいくつか説明する。

【 0 0 3 7 】

図 6 は、アプリケーション 1 3 が持つインターフェースの違いによってアプリケーション初期化・終了時処理実行部 8 が持つ 4 種のインターフェースを用いてワークフロー管理サーバとアプリケーションを統合した例である。なお、図 6 (a) のワークフロー管理サーバ 6 0 1、図 6 (b) のワークフロー管理サーバ 6 0 5、図 6 (c) のワークフロー管理サーバ 6 0 9、図 6 (d) のワークフロー管理サーバ 6 1 5 は、図 1 のワークフローサーバ 1 と同様のものである。また、図 6 (a) のサーバ接続部 6 0 2、図 6 (b) のサーバ接続部 6 0 6、図 6 (c

）のサーバ接続部 6 1 0、サーバ接続部 6 1 3、図 6（d）のサーバ接続部 6 1 6、サーバ接続部 6 1 9 は、図 1 のサーバ接続部 2 と同様のものである。さらに、図 6（a）のアプリケーション管理部 6 0 3、図 6（b）のアプリケーション管理部 6 0 7、図 6（c）のアプリケーション管理部 6 1 1、アプリケーション管理部 6 1 4、図 6（d）のアプリケーション管理部 6 1 7、アプリケーション管理部 6 2 0 は、図 1 のアプリケーション管理部 3 と同様のものである。

【0 0 3 8】

図 6（a）は、ワークフロー管理サーバ 1 が、アプリケーション 1 3 にサーバ接続部 2 を通して開始要求を送り、ワークフロー管理サーバ 1 はアプリケーション 1 3 の終了を待たずにプロセスを次作業に遷移させる統合の例（以下、「チェーン連携」という）である。ワークフロー管理サーバ 1 からアプリケーション 1 3 の開始要求がサーバ接続部 2 とアプリケーション初期化・終了時処理実行部 8 を通してアプリケーション 1 3 に送られる。この時、アプリケーション 1 3 の開始には、チェーン連携用の実装の雛型を利用してユーザが実装した初期化処理を利用する。アプリケーション管理部 3 は、アプリケーション 1 3 に開始要求を出したあと、アプリケーション 1 3 の終了を待たずに、サーバ接続部 2 を通してワークフロー管理サーバ 1 に実行終了の状態変更要求を出す。チェーン連携は、アプリケーション 1 3 が実行終了時の出力データを持たない場合に適用することが可能である。

【0 0 3 9】

図 6（b）は、ワークフロー管理サーバ 1 が、アプリケーション 1 3 にサーバ接続部 2 を通して開始要求を送り、その後アプリケーション 1 3 の終了を待ってプロセスを次作業に遷移させる統合の例である。このとき、ユーザはアプリケーション 1 3 固有の初期化処理と終了時処理 0 を一つの処理として実装する。このため、図 1 におけるアプリケーション管理部 3 内のアプリケーション状態通知部 7 は、アプリケーション 1 3 への実行開始要求後、ワークフロー管理サーバ 1 に実行開始済みの状態変更の要求を出さず、アプリケーション 1 3 の終了時処理後にワークフロー管理サーバ 1 に実行終了の状態変更要求を出す。これをネスト同期連携という。ネスト同期連携は、アプリケーション 1 3 が実行開始要求から終

了までの処理を一つのオペレーションで行うインターフェースを持つ場合や、アプリケーション 13 の実行開始から終了までの時間が短い場合に適用することが可能である。

【0040】

図 6 (c) は、ワークフロー管理サーバ 1 におけるプロセスの遷移のしかたはネスト同期連携と同じだが、アプリケーション 13 の終了時処理を別アダプタとして実装し、図 1 におけるアプリケーション初期化・終了時処理実行部 8 がアプリケーション 13 の終了をアプリケーション 13 に問い合わせることで、図 1 におけるアプリケーション状態通知部 7 に実行完了の要求を出す場合の例（以下、「ネスト非同期プル連携」という）である。ネスト非同期プル連携では、アプリケーション 13 への実行開始要求は、前記チェーン連携を用いて実現し、アプリケーション 13 への終了問い合わせは、図 1 におけるアプリケーション初期化・終了時処理実行部に、ユーザが問い合わせ用の雛型にしたがって実装した処理を用いて実現する。ネスト非同期プル連携は、アプリケーション 13 が終了を通知するインターフェースを持たない場合や、アプリケーション 13 の実行開始から終了までの時間が長い場合に適用することが可能である。

【0041】

図 6 (d) は、ワークフロー管理サーバ 1 におけるプロセスの遷移のしかたはネスト同期連携と同じだが、アプリケーション 13 の終了時処理を別アダプタとして実装し、アプリケーション 13 が、図 1 におけるアプリケーション初期化・終了時処理実行部 8 に終了を通知することで、図 1 におけるアプリケーション状態通知部 7 に実行完了の要求を出す場合の例（以下、「ネスト非同期プッシュ連携」という）である。ネスト非同期プッシュ連携では、アプリケーション 13 への実行開始要求は、前記チェーン連携を用いて実現し、アプリケーション 13 からの終了通知は、アプリケーション 13 が持つ固有のインターフェースと、図 1 におけるアプリケーション初期化・終了時処理実行部にユーザがプッシュ用の雛型にしたがって実装した処理とを用いて実現する。ネスト非同期プッシュ連携は、アプリケーション 13 が終了を通知するインターフェースを持っている場合や、アプリケーション 13 の実行開始から終了までの時間が長い場合に適用するこ

とが可能である。

【 0 0 4 2 】

図 7 は、業務プロセスの処理の一例を示したものである。本業務プロセスは、発注業務から部品調達業務までの簡単な例を示したものである。この処理を行うシステムの基本構成は図 1 に示したものと同一であり、パッケージアプリケーション起動・完了アダプタ 7 0 3、データベース登録用アダプタ 7 0 7 は図 1 のアプリケーション統合装置 1 4 に相当するものであり、基本的には同様の構成、処理機能を持つものである。パッケージアプリケーション 7 0 4 は、発注する商品の登録及び登録された商品の読み出しを行うためのものである。このパッケージアプリケーション 7 0 4 は、パッケージアプリケーション起動・完了アダプタ 7 0 3 を介して、ワークフロー管理サーバ 7 0 1 からの要求に基づいて起動・完了され、また図示しないユーザによって直接起動・完了ができるようになっている。また、部品調達アプリケーションは、入荷された製品を登録及び登録された商品の読み出しを行うためのものである。この部品調達アプリケーション 7 0 8 も、ワークフロー管理サーバ 7 0 1 からの要求に基づいて起動・完了され、また図示しないユーザによって直接起動・完了ができるようになっている。

【 0 0 4 3 】

図 8 はプロセスデータマッピング情報テーブルを示したものである。図 8 (a) はパッケージアプリケーション起動・完了アダプタ 7 0 3 で利用されるプロセスデータマッピング情報テーブルを示したものであり、図 8 (b) は部品調達アプリケーション 7 0 8 で利用されるプロセスデータマッピング情報テーブルを示したものである。図 8 (a) に示すようにパッケージアプリケーション 7 0 4 で扱うデータは、バイナリデータで表現された製品コード、個数、単価である。また、図 8 (b) に示すように部品調達アプリケーション 7 0 8 で扱うデータは、バイナリデータで表現された製品型名、個数及び合計金額である。

【 0 0 4 4 】

図 9 は、トランスレータマッピング情報テーブルを示したものである。図 9 (a) はパッケージアプリケーション起動・完了アダプタ 7 0 3 で利用されるトランスレータマッピング情報テーブルを示したものであり、図 9 (b) は部品調達

アプリケーション708で利用されるトランスレータマッピング情報テーブルを示したものである。ここでは、図9（b）に示したように部品調達アプリケーション708で扱う合計金額をパッケージアプリケーション704で扱う個数と単価で求めるようにしている。つまり、ここではデータの内容が異なるアプリケーション間の連携を実現している。

【0045】

以下、図7の処理について説明するが、基本的な処理フローは図2、図5、図11に示したものと同一である。

【0046】

部品の発注を行いたいユーザは、パッケージアプリケーションをそれぞれ直接起動し、発注する製品コード、個数、単価を登録する。

【0047】

一方、商品の発注、入荷作業を行うためにワークフロシステムが運用されると、ワークフロー管理サーバ701は発注作業を実行する。この発注作業は、ユーザによって登録された発注商品を確認する作業である。発注作業が実行されると、パッケージアプリケーション704に登録された商品の読み出しをパッケージアプリケーション起動・完了アダプタ703に要求する。パッケージアプリケーション起動完了アダプタ703は、図8（a）のプロセスデータマッピングテーブルを参照し、フローデータがあるか否かを確認する。ここでは、プロセスデータマッピングテーブルにフローデータが無いことが示されているので、パッケージアプリケーション起動・完了アダプタは、パッケージアプリケーション704を起動し、製品コード、個数、単価のデータ705を読み出す。全ての製品コード、個数、単価のデータの読み出しが完了するとパッケージアプリケーション704はパッケージアプリケーション起動・完了アダプタ703に終了通知する。パッケージアプリケーション起動・完了アダプタ703は、図9（a）に示すトランスレータマッピング情報テーブルから受け取ったデータ705の変換を行い、この変換されたデータと、ワークフロー管理サーバ701が発注作業の状態変更に必要なデータからプロセスデータ702を作成し、ワークフロー管理サーバ701へ受け渡す。ワークフロー管理サーバ701は発注作業の状態を完

了状態に変更し、次作業である入荷登録作業、検査完了作業を実行可能状態にする。入荷登録作業、検査完了作業は、プロセスデータを構成するデータである製品コード、個数、単価の条件を満たす入荷があった場合に、外部からの入力で、製品コードを製品型式に変換することで、製品型式、個数、合計金額のデータとしてデータベースに登録する作業を行う。つまり、製品型式に変換されることで、入荷、検査が完了したことになる。

【 0 0 4 8 】

入荷登録作業、検査完了作業が実行されたワークフロー管理サーバ 7 0 1 は、外部から製品形式の入力があると、製品型式、個数、合計金額の登録をデータベース登録用アダプタ 7 0 7 に要求する。

【 0 0 4 9 】

要求を受けたデータベース登録用アダプタ 7 0 7 は、図 8 (b) のプロセスデータマッピング情報テーブルを参照し、フローデータがあるかを確認する。ここでは、プロセスデータマッピング情報テーブルにフローデータが有ることが示されているので、プロセスデータからフローデータを取り出す。ここでは、製品形式、個数、単価がフローデータとなる。次に、図 9 (b) に示すトランスレータマッピング情報テーブルを参照し、フローデータを変換する必要があるか判断する。ここでは、変換する必要があることが示されているので、トランスレータマッピング情報テーブルを利用して、フローデータをアプリケーション固有のデータに変換する。ここでは、個数と単価から合計金額とする変換を行う。次に、データベース登録用アダプタは、部品調達アプリケーションを起動し、製品形式、個数、合計金額を登録する。

【 0 0 5 0 】

部品調達アプリケーション 7 0 8 の処理が終了すると、データベース登録用アダプタ 7 0 7 は、ワークフロー管理サーバ 7 0 1 に入荷登録作業、検査完了作業の状態を完了状態に変更する要求を出す。要求を受け取ったワークフロー管理サーバ 7 0 1 は入荷登録作業、検査完了作業の状態を完了状態に変更する。

【 0 0 5 1 】

このように、本発明によれば、データの内容が異なる複数のアプリケーション

を相互に実行可能なシステムを構築することが可能となる。

【 0 0 5 2 】

第 1 のアプリケーションの実行結果を第 2 のアプリケーションで利用可能とするために変換し、変換した実行結果を利用して第 2 のアプリケーションを実行することで、データの形式又は内容が異なるアプリケーションの実行が可能となる。

【 0 0 5 3 】

更に、第 1 のアプリケーションの実行結果を変換するためには、第 2 のアプリケーションとのデータの項目と、形式又は／及び内容を定義すれば良い。その他は、アプリケーションで共通に利用可能である。

【 0 0 5 4 】

また、アプリケーションを実行させるためには、アプリケーションの初期化処理、終了時処理を定義すれば良い。その他は、アプリケーションで共通に利用可能である。

【 0 0 5 5 】

【発明の効果】

本発明によれば、既存のアプリケーションを有効に利用可能なシステムを構築することが可能となる。

【図面の簡単な説明】

【図 1】

本発明を適用したワークフローシステムの構成を示した図である。

【図 2】

サーバ接続部の処理を示した図である。

【図 3】

プロセスデータマッピング情報テーブルを示した図である。

【図 4】

トランスレータマッピング情報テーブルを示した図である。

【図 5】

アプリケーション管理部の処理を示した図である。

【図 6】

ワークフロー管理サーバとアプリケーションとの統合の一例を示した図である。

【図 7】

業務プロセスの一例を示した図である。

【図 8】

図 7 の業務プロセスを行うためのプロセスデータマッピング情報テーブルの一例を示した図である。

【図 9】

図 7 の業務プロセスを行うためのトランスレータマッピング情報テーブルの一例を示した図である。

【図 1 0】

本発明の基本構成を示した図である。

【図 1 1】

サーバ接続の処理を示した図である。

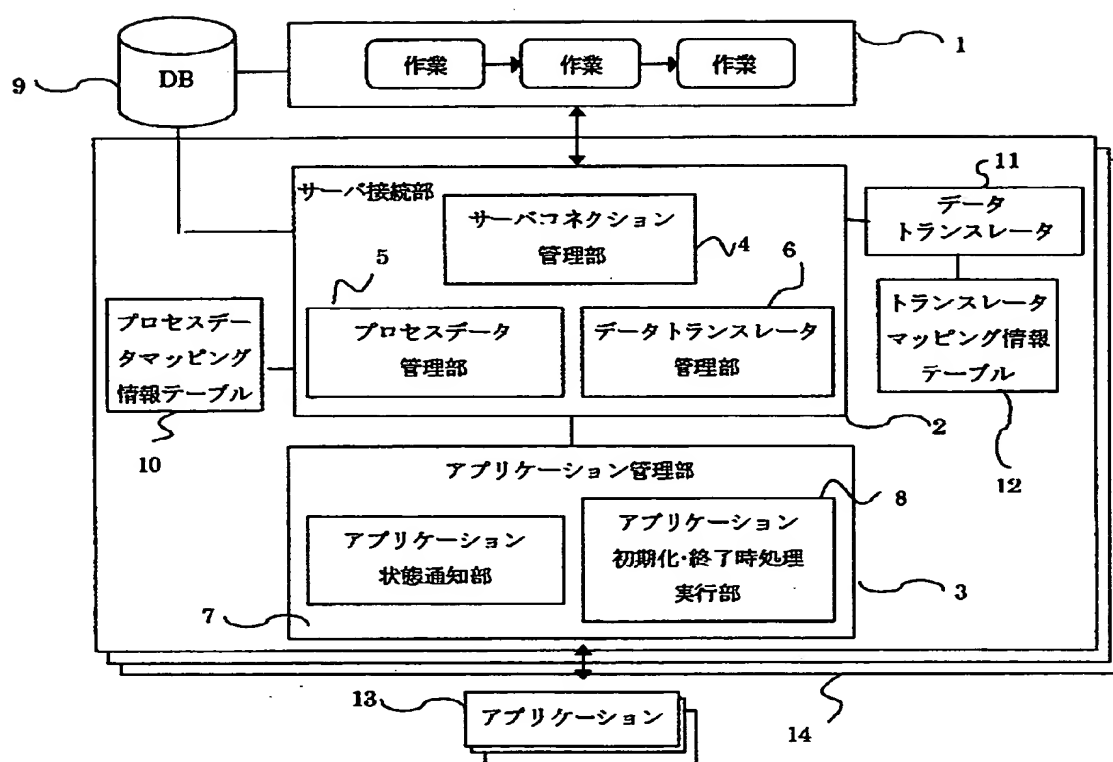
【符号の説明】

2・・・サーバ接続部、 3・・・アプリケーション管理部、 4・・・サーバコネクション管理部、 5・・・プロセスデータ管理部、 6・・・データトランスレータ管理部、 7・・・アプリケーション状態通知部、 8・・・アプリケーション初期化・終了時処理実行部。

【書類名】 図面

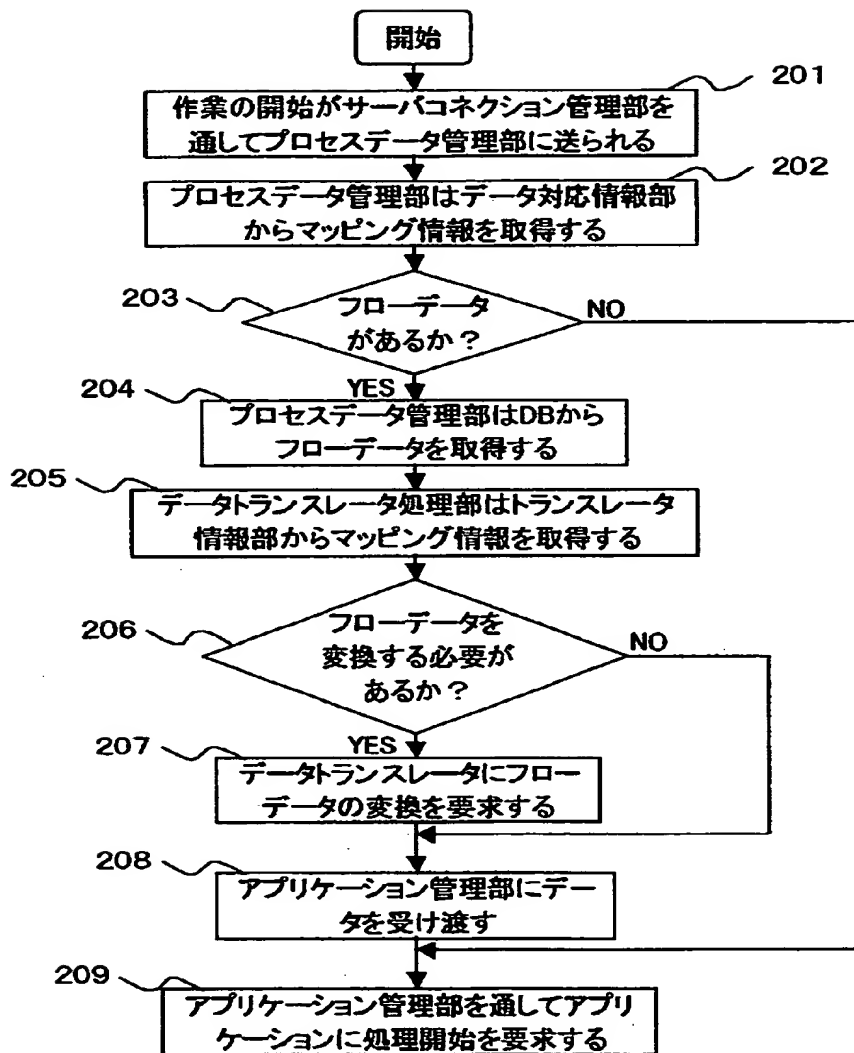
【図 1】

図 1



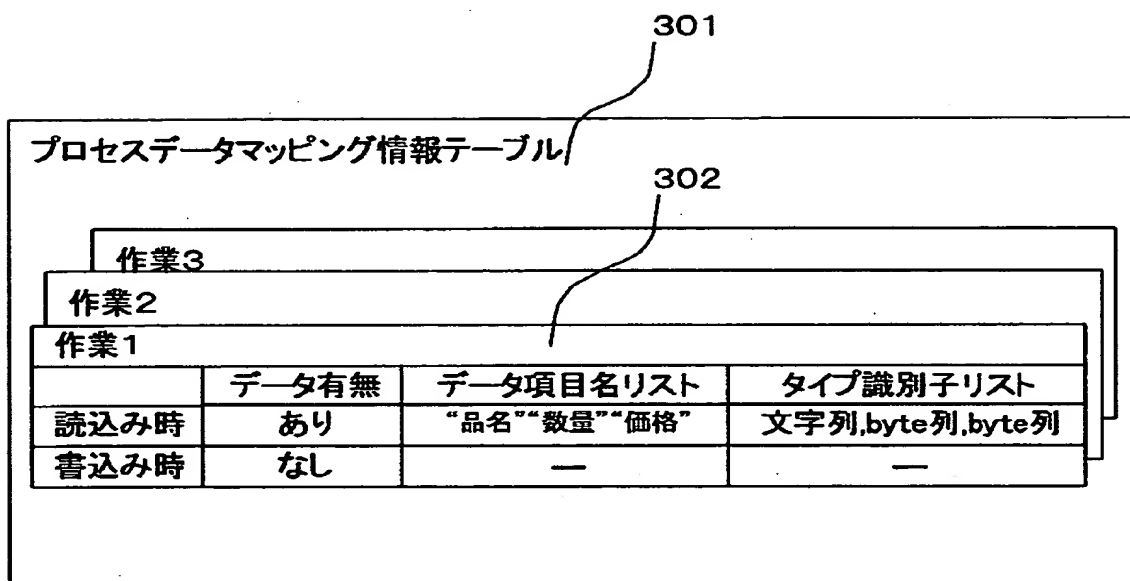
【図 2】

図 2



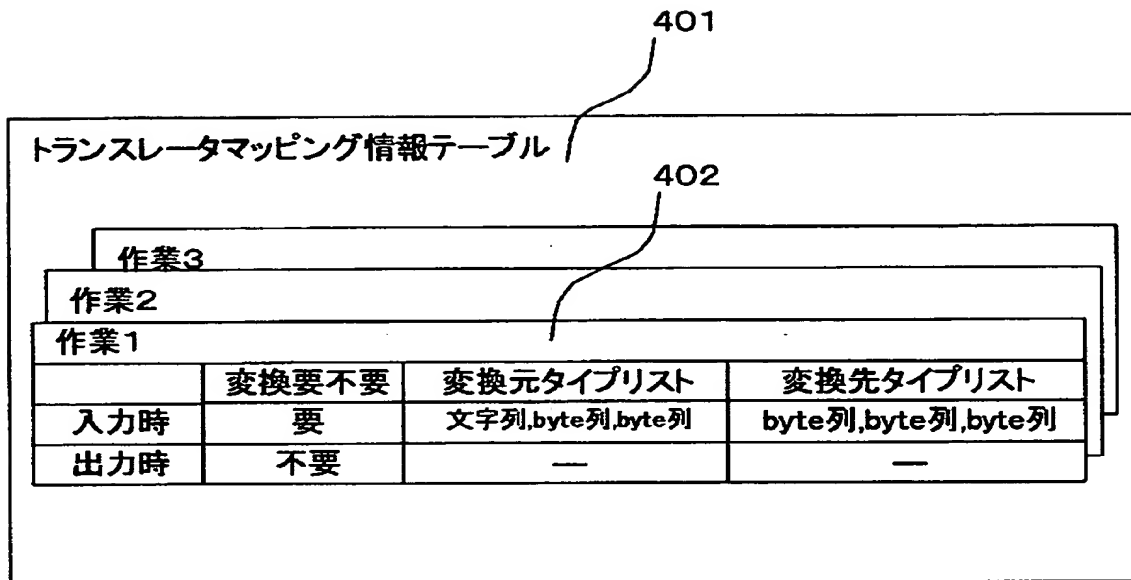
【図 3】

図 3



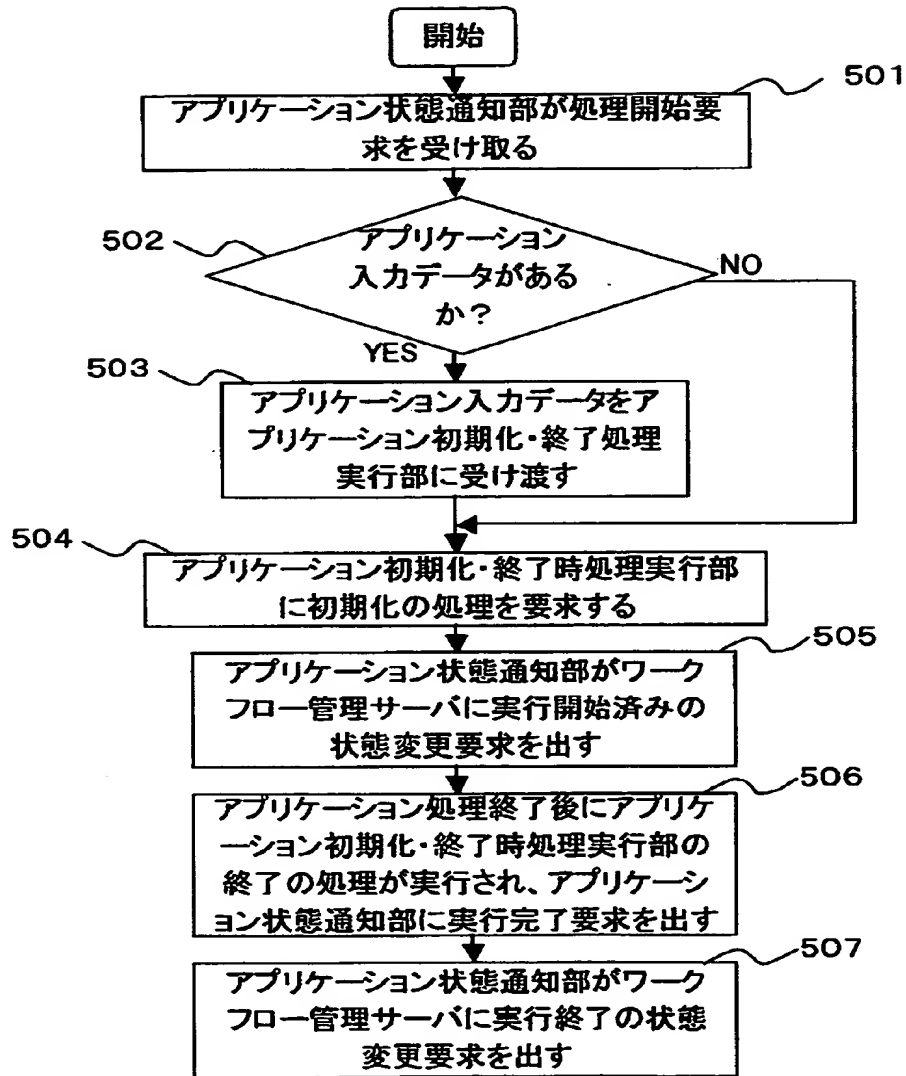
【図 4】

図 4



【図 5】

図 5



【図 6】

図 6 (a)

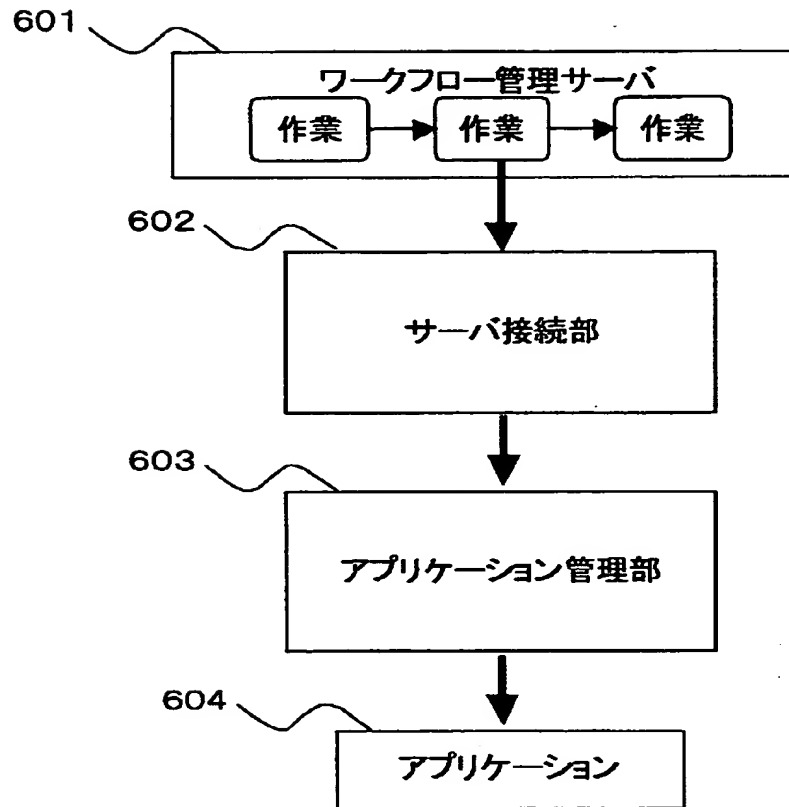


図 6 (b)

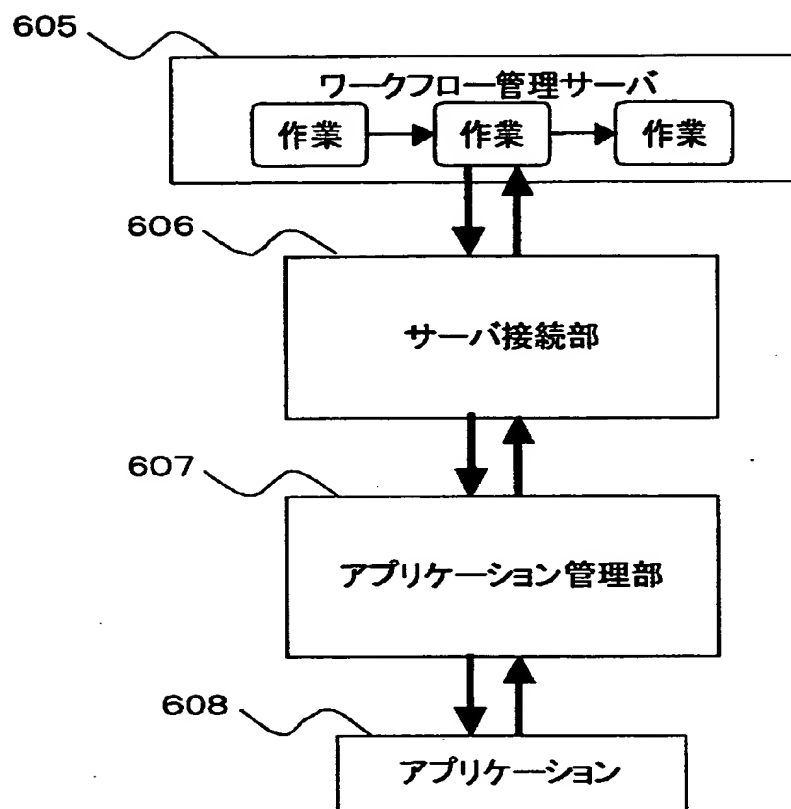


図 6 (c)

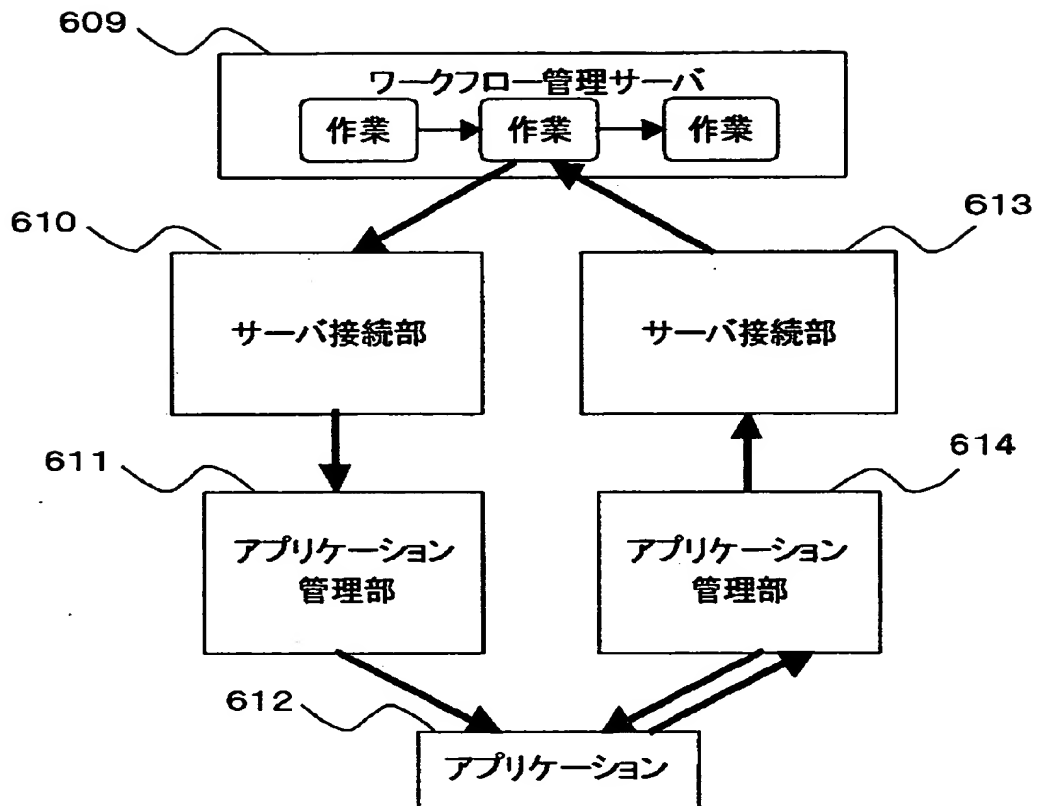
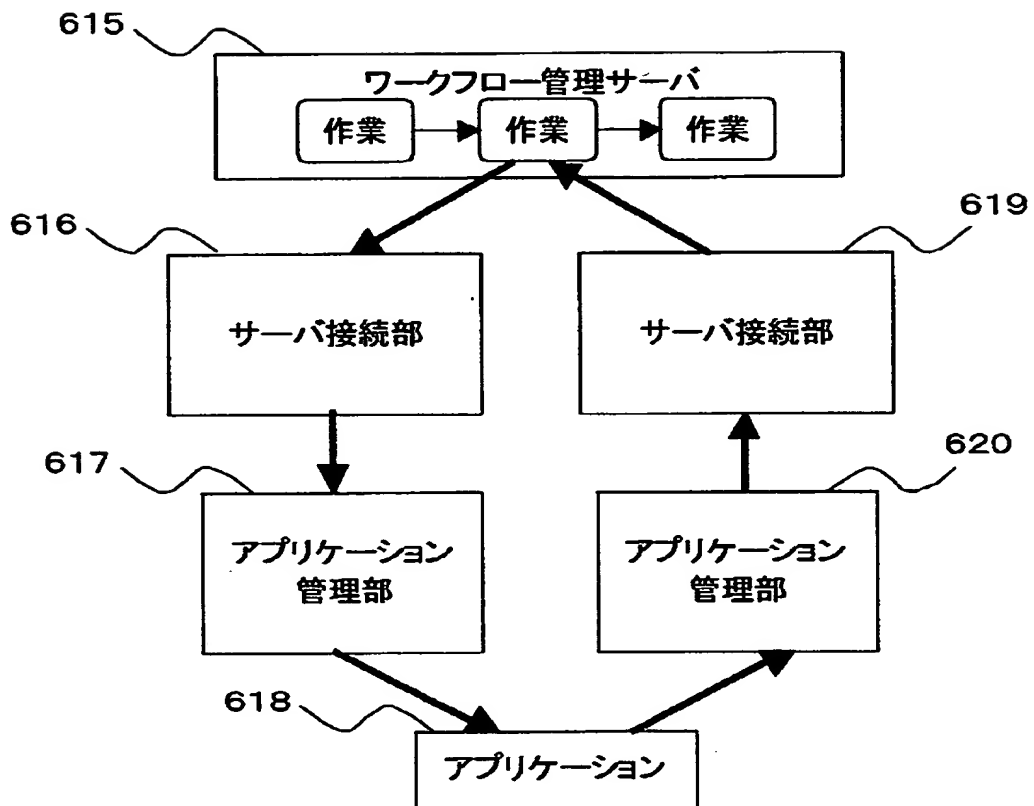
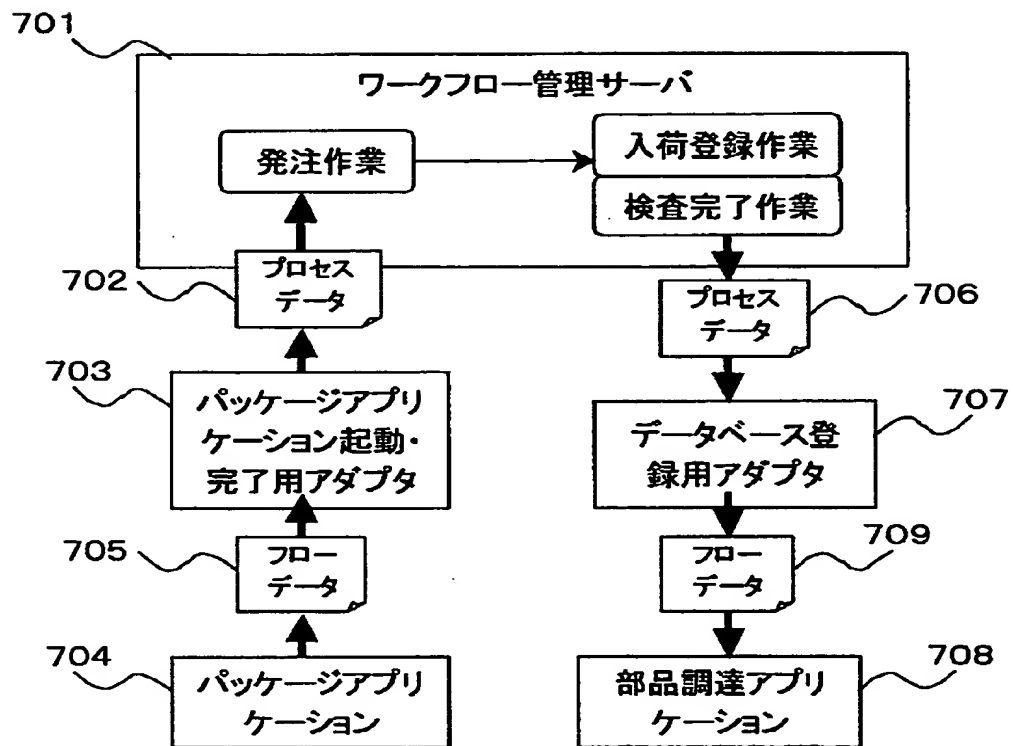


図 6 (d)



【図 7】

図 7



【図 8】

図 8

(a)

| | データ有無 | データ項目名リスト | タイプ識別子リスト |
|-----|-------|-------------|----------------------|
| 読込時 | あり | 製品コード、個数、単価 | byte 列、byte 列、byte 列 |
| 書込時 | なし | --- | — |

(b)

| | データ有無 | データ項目名リスト | タイプ識別子リスト |
|-----|-------|------------|----------------------|
| 読込時 | なし | | |
| 書込時 | あり | 製品形式、個数、単価 | byte 列、byte 列、byte 列 |

【図 9】

図 9
(a)

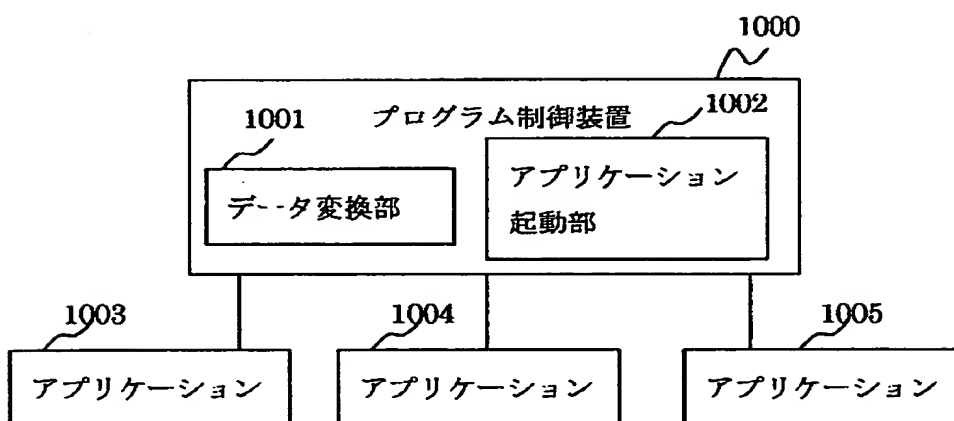
| | 変換要不要 | 変換元タイプリスト | 変換先タイプリスト |
|-----|-------|----------------------|----------------------|
| 入力時 | 不要 | — | — |
| 出力時 | 要 | byte 列、byte 列、byte 列 | byte 列、byte 列、byte 列 |

(b)

| | 変換要不要 | 変換元タイプリスト | 変換先タイプリスト |
|-----|-------|----------------------|--------------------------------------|
| 入力時 | 要 | byte 列、byte 列、byte 列 | byte 列、byte 列、byte 列 — 、 — 、個数×単価 |
| 出力時 | 不要 | — | — |

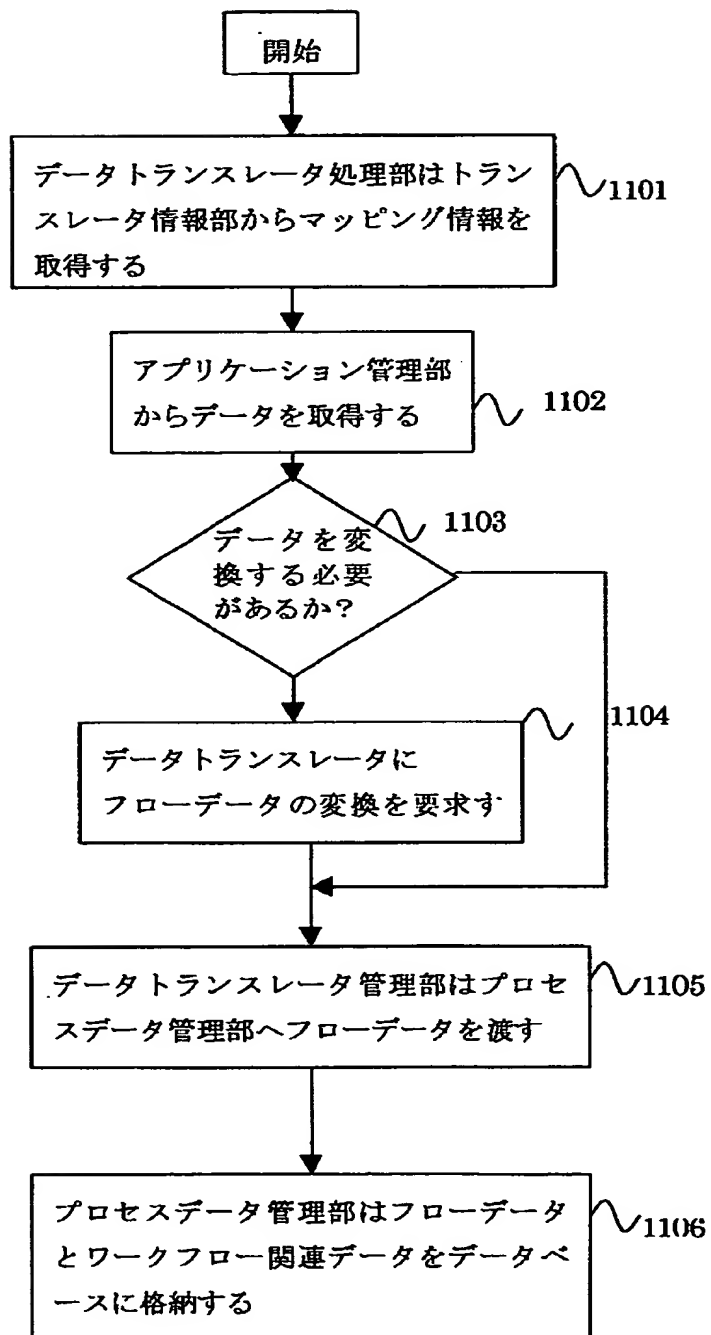
【図 1 0】

図 1 0



【図11】

図11



【書類名】 要約書

【要約】

【課題】

複数のアプリケーションを利用して処理を行うシステム、方法を提供する。

【解決手段】

アプリケーションで利用されるデータの対応関係を予め定義しておき、データ変換部 1 0 0 1 で、アプリケーションの実行結果のデータを変換する。変換されたデータを利用してアプリケーション起動部は、別のアプリケーションを実行する。

【選択図】 図 1 0

出 願 人 履 歴 情 報

識別番号 [000005108]

1. 変更年月日 1990年 8月31日

[変更理由] 新規登録

住 所 東京都千代田区神田駿河台4丁目6番地
氏 名 株式会社日立製作所